

## **METHOD FOR RECONSTRUCTION OF PIXEL COLOR VALUES**

### **BACKGROUND OF THE INVENTION**

The priority benefits of the October 10, 2002 filing dates of U.S. provisional applications serial nos. 60/417,142 and 60/417,152 are hereby claimed.

#### **5    Field of the Invention**

The present invention relates to a method and apparatus for object-based color reconstruction in a multi-color based sensor arrangement through maintaining coherence within objects and along edges, and a computer program and computer program product for controlling such method, and an image sensing facility comprising such apparatus. In particular, the invention relates to a method for executing object-based color reconstruction  
10    in a multi-color matrix-based sensor arrangement that estimates color value through determining local gradients among various luminance component values assessed.

#### **Description Of Related Art**

15        Multi-color matrix-based sensor arrangements that include color sensors that have one first luminance component sensed at a relatively higher spatial frequency and two further chrominance components sensed at relatively lower spatial frequencies are coming in rapidly expanding use. Fields of application include digital cameras, digital cinematography, video cameras, scientific imaging, broadcast cameras, surveillance, security monitoring, and various  
20    others. Now generally, each object point translates into a single-color of the corresponding image pixel. Inasmuch as, in the resulting image each pixel lacks two of the three sensor colors, reconstruction of three-color pixels requires an appreciable amount of processing.

However, in earlier realizations, the resulting reconstructed array often had artificially colored, noise-enhanced, or blurred edges. In consequence, there exists a growing need to provide for an algorithmic procedure that will provide high-quality estimations for the color values of pixels that were originally sensed in a single color only, whilst requiring to access  
5 only a minimal memory facility and minimal power consumption, needing to access only a small number of image lines/columns at a time, and being independent of other DSP-based (i.e., Digital Signal Processing-based) algorithms, such as those for removing false colors or correcting color balance, overall influencing of hue or luminance, or others. As will be discussed more extensively hereinafter, the present embodiment generally utilizes only a  
10 small data kernel of optimally 5 x 5 pixels for Green, or minimally 3 x 3 pixels for Red and Blue. It will generally minimize perceptual error, it reconstructs frequencies above the Nyquist frequency, it has much lower color aliasing than many other algorithms, it will hardly expand existing defects, and it will only be marginally affected by image noise.

United States Patent Application Publication 2002/0063789 A1 to Acharya et al and  
15 published on May 30, 2002 discloses a Color Filter Array and Color Interpolation Algorithm, and uses a classification to determine which pixels it will use in the interpolation. The present invention in contradistinction uses a soft decision method that is useful for adaptive adjusting for variations in system noise. Moreover, the reference singles out a very particular matrix design with a very high percentage of 75% Green pixels sensed. The present invention is  
20 much wider applicable.

Moreover, the inventor has found that correct application of the algorithm will not remove fundamental signal noise. Rather, it will prevent enhancing of the noise. In particular, the inventor has recognized the corresponding behavior of the gradients among the various sensed colors.  
25

## SUMMARY OF THE INVENTION

In consequence, amongst other things, it is an object of the present invention to use such corresponding behavior to attain an improved result against an investment of only a limited amount of processing complexity.

30 By itself, co-pending patent application by the present inventor, titled "Method And Apparatus For Adaptive Pixel Correction Of A Multi-Color Matrix," identified by Attorney

Docket No. 12546 and assigned to the same assignee as the present application relates to the adaptive correcting of defective pixels and is incorporated herein by reference. The adaptive pixel correction of a multi-color matrix is based on soft decisions across the various color planes, and in particular based on taking into account the spatial response pattern of the human visual system. The result of the co-pending application may be taken as a starting point for applying the present invention, but such represents no express restriction.

A method for object-based color reconstruction in a multi-color matrix-based sensor arrangement is based on color sensors that have one first color sensed at a relatively higher spatial frequency and two further colors sensed at relatively lower spatial frequencies.

10 In particular, the method executes the following steps. A particular pixel *not* sensed in said first color has its first color value estimated through determining local gradients (82) among various first color values assessed, and in accordance with such gradients such estimating is executed through interpolating (84, 86) along relatively stronger edge informations. For a particular pixel *not* sensed in a particular further color that further color's value is estimated in a direction along with relatively smaller differences evaluated in the first color (94, 96). Advantageously, the method has the ability for limiting noise propagation.

Now therefore, according to one of its aspects the invention a method includes a first process for a first pixel and a second process for the first pixel. The first process includes extracting a first kernel from a multi-color matrix, generating first variance weights from the first kernel, and generating a first color based on the first variance weights and adjacent pixel values of the first color. The second process includes extracting a second kernel from the multi-color matrix, generating second variance offsets from the second kernel, and generating a second color based on the second variance offsets and an adjacent pixel of the second color.

The invention also relates to a computing machine that includes a processor and a memory. The memory stores a first module for controlling the processor to perform a first process for a first pixel. The first process includes extracting a first kernel from a multi-color matrix, generating first variance weights from the first kernel, and generating a first color based on the first variance weights and adjacent pixel values of the first color. The memory further stores a second module for controlling the processor to perform a second process for the first pixel. The second process includes extracting a second kernel from the multi-color

matrix, generating second variance offsets from the second kernel, and generating a second color based on the second variance offsets and an adjacent pixel of the second color.

The invention also relates to a computer readable medium having stored thereon a plurality of modules for controlling a processor. The plurality of modules includes a first module and a second module. The first module is for controlling the processor to perform a first process for the first pixel. The first process includes extracting a first kernel from a multi-color matrix, generating first variance weights from the first kernel, and generating a first color based on the first variance weights and adjacent pixel values of the first color. The second module is for controlling the processor to perform a second process for the first pixel. The second process includes extracting a second kernel from the multi-color matrix, generating second variance offsets from the second kernel, and generating a second color based on the second variance offsets and an adjacent pixel of the second color.

Further advantageous aspects of the invention are as disclosed and claimed herein.

## **BRIEF DESCRIPTION OF DRAWINGS**

The invention will be described in detail in the following description of preferred embodiments with reference to the following figures wherein:

FIG. 1 is a schematic view of a Bayer matrix of image points;

FIG. 2 is a schematic view of the deriving of the horizontal Green gradient;

FIG. 3 is a schematic view of the deriving of the vertical Green gradient;

FIGS. 4a, 4b are schematic views of the relevance of the eight possible *Green* directions;

FIG. 5 is a graph of a filter characteristic diagram for removing false color responses; and

FIG. 6 is a schematic block diagram of a hardware implementation of the present invention.

## **DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS**

FIG. 1 illustrates a Bayer matrix of image points for use with the present invention. Alternatively, other arrangements could be used, such as through using subtractive colors, such as cyan, magenta, and yellow, or through various other published embodiments. As

shown, the matrix has Green pixels arranged according to a chessboard organization. The remaining pixels have been used for Red pixels in odd rows and Blue pixels in even rows.

Now, the intended end result of the reconstruction embodiment is three complete images of Red, Green, and Blue pixel values, respectively, where the original recording did  
 5 yield less than three values for each respective pixel, according to the pixel pattern for the respective single color. It is known from advance that at each pixel location, two values must be estimated. In practice, the application most favored is to a Bayer pattern wherein the green pixels form a chessboard pattern, while the non-green fields have alternating red and blue values in the diagonal directions.

10 Therefore, first the Green matrix will be completed. In this respect, FIG. 2 illustrates the deriving of the horizontal Green gradient or horizontal green edge indicator  $\delta_H$  in a preferred kernel pattern of 5 x 5 pixels according to the average absolute differences between pixels pairs joined by an arrow, according to:

15 
$$\delta_H = \{ |G_{i-1,j-2} - G_{i-1,j}| + |G_{i-1,j} - G_{i-1,j+2}| + \dots \} / 5.$$

Likewise, FIG. 3 illustrates the deriving of the vertical Green gradient or vertical Green edge indicator  $\delta_V$ . For the time being, the pixels assigned to other colors will be ignored. In certain alternative environments, it may be advantageous to define the kernel as  
 20 having a size of 3 x 3 pixels only, through using the two arrows numbered 23 in FIG. 2, and 27 in FIG. 3, respectively. Apparently, the respective kernels for horizontal and vertical determinations will not anymore coincide here.

Then, in a relatively simplistic implementation a local edge direction is determined and a Green value for the missing central point is estimated as follows: if  $\delta_V$  is relatively  
 25 larger, interpolate horizontally between two nearest green neighbors; if  $\delta_H$  is relatively larger, interpolate vertically between two nearest green neighbors, and if  $\delta_H$  and  $\delta_V$  are roughly equal, interpolate linearly between all four nearest neighbors of the missing Green pixel. Of course, the determining of the being “roughly” equal allows to introduce a certain amount of flexibility into the algorithm. The logical method presented here demonstrates the functioning  
 30 of the actual algebraic method.

A fuller realization as an algebraic expression for the interpolated green pixel is:

$$G_{i,j} = \frac{1}{2} \lambda_H (G_{i,j-1} + G_{i,j+1}) + \frac{1}{2} \lambda_V (G_{i-1,j} + G_{i+1,j}),$$

5 wherein  $\lambda_H = 1 - \delta_H^k / (\delta_H^k + \delta_V^k)$ , and likewise, for the vertical parameter. Herein, a greater value of exponent constant  $k$  increases the algorithm's sensitivity to the differences in  $\delta_H$  and  $\delta_V$ .

The algebraic expressions offer more benefits than just for presentation. The non-linear combination of the edge indicators can be adjusted with the value of  $k$  to either  
 10 emphasize edge clarity in low-noise situations, or rather avoid the enhancing of the noise while still maintaining minimal color aliasing in high-noise situations. Therefore,  $k$  has a two-sided adaptive effect on the algorithm. With small values (between 1 and 2) noise will be adaptively minimized whilst ensuring that edges will have none to minimal color aliasing. On the other hand, with larger values for  $k$  in low-noise situations, more emphasis can be put on  
 15 removing all color aliasing whilst enhancing edge continuity and sharpness. Further, the use of five arrows in FIGS. 2 and 3, instead of the minimal number of one, allows reconstruction of frequencies up to twice the Nyquist frequency  $f_N$  (i.e. the spatial sample frequency) in both vertical and horizontal directions. In case of the smaller kernels, the limit would lie at the Nyquist frequency.

20 After having completed the Green array  $G_{i,j}$ , the missing points in the Red and Blue images will be estimated. For a Green sample on a Green-Blue row, the horizontal neighbors are Blue and the vertical neighbors are Red. First the Red value is sought for. To ensure the *continuity of the object* recorded at the given point, the vertically *closest* Green value to the actual pixel is determined and then its associated Red value copied to the actual pixel. For  
 25 Blue, among the two neighbors, the horizontally *closest* Green value to the actual pixel is determined and then its associated Blue value copied to the actual pixel.

For a Blue sample, one approach is to choose among the four diagonally adjacent Red values, whilst using the respective Green differences shown in FIG. 4b, as follows:

30 
$$R_{i,j} = (R_{i-1,j+1} - G_{i-1,j+1} + G_{i,j}) \text{ if } \Delta g_2 = \Delta_{\min}$$



i.e. add the minimum Green difference to the Red value, and correspondingly:

$$\begin{aligned}
 & \mathbf{R}_{i,j} = ( \mathbf{R}_{i+1,j+1} - \mathbf{G}_{i+1,j+1} + \mathbf{G}_{i,j} ) \text{ if } \Delta g_4 = \Delta_{\min} \\
 & \mathbf{R}_{i,j} = ( \mathbf{R}_{i+1,j-1} - \mathbf{G}_{i+1,j-1} + \mathbf{G}_{i,j} ) \text{ if } \Delta g_6 = \Delta_{\min} \\
 & \mathbf{R}_{i,j} = ( \mathbf{R}_{i-1,j-1} - \mathbf{G}_{i-1,j-1} + \mathbf{G}_{i,j} ) \text{ if } \Delta g_8 = \Delta_{\min}
 \end{aligned}$$

A corresponding version applies to the missing Blue values at pixels that were sampled Red.

Now, although the color reconstruction according to the present invention's embodiment will generally be correct, the particular arrangement of the Bayer matrix, in combination with non-ideal optical image low-pass filtering and system noise can cause some false color responses. Heuristically, the following description is given. The sampling of a scene through a Bayer matrix will cause two major sampling errors, as follows: (a) The relative displacement between green, red and blue samples introduces a phase error between the various colors; (b) The sampling frequencies of Red and Blue are relatively lower than that of Green. This is perceptually advantageous, because the human eye's chroma and hue resolution is relatively less than its luminance resolution. On the other hand, this introduces different aliasing responses for Green versus for Red/Blue, in particular, inasmuch as Green will attain a better high-frequency response. Although this may cause false-color spikes and the like, such phenomenon is relatively rare. Such spikes can be considered high-frequency chromatic points, and low-pass filtering would generally remove them. In fact, a chroma image can be estimated through two Green differences

$$\mathbf{C}' = ( \mathbf{R} - \mathbf{G} )_{i,j}, \text{ and } \mathbf{C}'' = ( \mathbf{B} - \mathbf{G} )_{i,j}.$$

In this respect, FIG. 5 shows the intended frequency response of a chroma filter. The design of such  $\alpha$ -filter that has been shown in solid lines is dependent on the system's optical low-pass filtering as well as on system noise. Ideally, it will be designed to minimize the loss of image information and at the same time maximize the removal of false colors. To an appreciable extent, degradation in sharpness through the alpha-filter is mitigated through a beta-filter shown in FIG. 5 in dashed lines.

FIG. 6 illustrates a hardware implementation of the present invention. Now, the output of the co-pending application referred to supra may be fed to a DSP pipeline 20 for processing and/or transient storage, as the case may be. The DSP pipeline is fed by the image sensing facility, not shown for brevity, whereas the pipeline itself has a sufficient amount of memory and processing facilities for implementing the method according to the present invention. In addition thereto, in block 70, the system will evaluate the noise level in the image, and therefrom in block 72 convert the noise level in the  $k$ -factor considered supra. From a holding element 74, this  $k$ -factor is provided to such elements of the hardware as are in need of it. Furthermore, for simplicity, command, control and synchronization streams have been left out of the embodiment for brevity and optimum clarification.

Now, block 22 represents the captured Bayer image information. In block 60, it is detected whether the pixel in question is at least 2 pixels away from the image edge. If not, the procedure hereinafter will not have enough data available, so the process will then move to the next pixel in block 62. In block 64, the system detects whether the sample pixel is Green.

If not, the system will undertake to reconstruct an appropriate Green value for the pixel in question, as follows. First, in block 80, the system extracts in a kernel of 5 x 5 pixels disclosed earlier the sensed Green pixels. In block 82, the horizontal and vertical variance parameters are derived. In block 84, the horizontal and vertical weight generating functions are derived, through also receiving the actual value of the exponent  $k$  from block 74, as discussed earlier. Finally, in block 86, the system calculates a product of the sum weights and the horizontally and vertically sampled Green pixels. With the so reconstructed Green pixel, the system then proceeds to mixer/accumulator 112 whilst in block 114 storing the result and in block 116 detecting whether the end of the image had been reached, which for the time being is not so.

If in block 64 the sample is not Green, so in block 66 the system shifts by one row and one column and checks for a Red pixel. If no, the pixel thus being Blue, the system will undertake to reconstruct a Red pixel. First, in block 90, the Red pixels are extracted in a 3 x 3 kernel as shown earlier, as well as the Green pixels corresponding thereto. Next, in block 92, the system generates object-based parameters for sampled Red pixels in the kernel. Next, in block in block 94, the system generates weights for the object-based parameter, whilst again



using the actual value of exponent  $k$  received from block 74. Finally, in block 96, a sum of the products of weights and measured Red pixels in the kernel generates the Red pixel value. Finally, in block 110, the system checks whether the *sample* pixel was Blue. If yes, the way to mixer facility 112 lies open. If no, the system reverts to the blue reconstructing column further to the right. The various blocks 100 through 106 for this column correspond to their respective counterparts in the middle column, and further extensive discussion thereof is omitted. This column is also entered from block 68 if the sampled pixel were Red itself.

Generally, after numerous rounds through the reconstructing columns, the answer in block 116 will become yes. This means that in block 118 the full RGB image will have been recovered; the remainder of the flow-chart implements a few largely cosmetic refinements. In block 120, the image is converted to an estimated color space with a chroma component applied. In block 122, a low pass chroma filter is used to substantially remove false colors. In block 124, the system executes a back conversion to the RGB color space. This will in block 126 yield the reconstructed RGB image that is subsequently presented to the DSP pipeline for further usage.

As an example of the above described embodiment, a method may be regarded as having first and second processes for a first pixel. For example, in the Bayer patterned multi-color matrix, if the first pixel is a Blue pixel, then the first process computes a Green value for the pixel and the second process computes a Red value for the pixel. The first process includes extracting a first kernel from a multi-color matrix. In the example discussed herein, the first kernel is a 5 by 5 sub matrix of the Bayer pattern centered on the exemplary Blue pixel. The first process further includes generating first variance weights (e.g.,  $\lambda_H$ ,  $\lambda_V$ ) from the first kernel and then generating a first color (Green, in this example) based on the first variance weights and adjacent pixel values of the first color.

The generating of the first variance weights includes first determining horizontal and vertical gradient value averages (e.g.,  $\delta_H$  and  $\delta_V$ ). The center of the first kernel is the exemplary Blue pixel, and the kernel includes first and second side pixels of the first color spaced apart horizontally. In this example, the first color is Green, and the first and second side pixels are Green pixels of the kernel that are spaced apart horizontally. To determine the horizontal gradient value average (e.g.,  $\delta_H$ ), the first process determines plural horizontal

gradients (e.g., pairs of two horizontally spaced Green pixels) and then determines a gradient value for each gradient. A first horizontal gradient value is determined by calculating an absolute value of a difference between a value of the first side pixel (a Green pixel in this example) and a value of the second side pixel (another Green pixel in this example). The calculation of the horizontal gradient value is repeated for each horizontal gradient within the first kernel. Then, the first process calculates an average of the plural horizontal gradient values to determine the horizontal gradient value average (e.g.,  $\delta_H$ ). For example, five horizontal gradient values are determined for the gradients depicted in FIG. 2. Then, the average is determined by summing the values and dividing by 5. The vertical gradient value average (e.g.,  $\delta_V$ ) is computed in a similar way. See FIG. 3.

The first variance weights include horizontal and vertical interpolation weights (e.g.,  $\lambda_H$ ,  $\lambda_V$ ). The first process determines the horizontal and vertical interpolation weights based on the horizontal and vertical gradient value averages and a predetermined exponent value,  $k$ . The horizontal interpolation weight is calculated based on

$$\lambda_H = 1 - \delta_H^k / (\delta_H^k + \delta_V^k), \text{ and}$$

the vertical interpolation weight is calculated based on

$$\lambda_V = 1 - \delta_V^k / (\delta_H^k + \delta_V^k),$$

where  $\delta_H^k$  is the horizontal gradient value average,  $\delta_V^k$  is the vertical gradient value average, and  $k$  is the predetermined exponent value.

The second process includes extracting a second kernel from the multi-color matrix. In the example discussed herein, the second kernel is a 3 by 3 sub matrix of the Bayer pattern centered on the exemplary Blue pixel. The second process further includes generating second variance offsets

$$(\text{e.g., } \Delta g_2 = G_{i-1,j+1} - G_{ij}, \Delta g_4 = G_{i+1,j+1} - G_{ij}, \Delta g_6 = G_{i+1,j-1} - G_{ij}, \text{ and } \Delta g_8 = G_{i-1,j-1} - G_{ij}).$$

The second variance offsets are generated from the second kernel. In the example discussed herein, the second kernel is a 3 by 3 sub matrix of the Bayer pattern centered on the exemplary Blue pixel. As discussed with respect to the first process, a Green value has been

calculated to become associated with the exemplary Blue pixel, and associated with each of the four Red pixels at the four corners of the second kernel, as discussed herein. Also see FIG. 1. With these calculated Green values, second variance offsets (e.g.,  $\Delta g_2$ ,  $\Delta g_4$ ,  $\Delta g_6$ , and  $\Delta g_8$ ) are determined based on the respective differences between the calculated Green values as depicted in FIGS. 4a and 4b. Then, the second color (e.g., Red in this example) is generated based on the second variance offsets and an adjacent pixel of the second color.

More specifically, generating the second variance offsets includes determining diagonal gradients between pixels in the second kernel (e.g.,  $[i, j]$  to  $[i-1, j+1]$ ,  $[i, j]$  to  $[i+1, j+1]$ ,  $[i, j]$  to  $[i+1, j-1]$  and  $[i, j]$  to  $[i-1, j-1]$ ) and then determining gradient values of the first color (Green in this example)

$$(e.g., \Delta g_2 = G_{i-1, j+1} - G_{ij}, \Delta g_4 = G_{i+1, j+1} - G_{ij}, \Delta g_6 = G_{i+1, j-1} - G_{ij}, \text{ and } \Delta g_8 = G_{i-1, j-1} - G_{ij})$$

corresponding to the gradients.

Then, generating a second color (Red, in this example) includes choosing a minimum value of the gradient values (e.g.,  $\Delta g_2$ ), selecting the adjacent pixel (e.g.,  $[i-1, j+1]$  to the right and above the center of the second kernel as depicted in FIG. 4b) of the second color (Red, in this example) based on a gradient that corresponds to the minimum value, and subtracting the minimum value (e.g.,  $\Delta g_2$ ) from a pixel value (e.g.,  $R_{i-1, j+1}$ ) of the selected adjacent pixel to determine the value for the second color (e.g.,  $R_{ij}$ ) at the center of the second kernel (e.g., the exemplary Blue pixel).

However, if  $\Delta g_4 = \Delta_{min}$  then,

$$R_{ij} = (R_{i+1, j+1} - G_{i+1, j+1} + G_{ij}),$$

or if  $\Delta g_6 = \Delta_{min}$  then,

$$R_{ij} = (R_{i+1, j-1} - G_{i+1, j-1} + G_{ij}),$$

or if  $\Delta g_8 = \Delta_{min}$  then,

$$R_{ij} = (R_{i-1, j-1} - G_{i-1, j-1} + G_{ij}).$$

On the other hand, if the center of the first and second kernels were a Red pixel, the second process would determine a Blue value for the center pixel.

The method repeats for every pixel (except pixels on the edge of the matrix where a full kernel cannot be extracted) so as to determine all three color planes for each pixel. For a second pixel, the method further includes a third process. The third process includes extracting a third kernel (e.g., a 3 by 3 sub matrix centered on the second pixel) from the multi-color matrix (e.g., the Bayer multi-color matrix described in this example). The third process further includes generating third variance offsets from the third kernel (e.g.,  $\Delta g_2$ ,  $\Delta g_4$ ,  $\Delta g_6$ , and  $\Delta g_8$ ) in a process substantially the same as is described above with respect to the second process, and generating a third color (Blue in this example) based on the third variance offsets and an adjacent pixel of the third color in a process substantially the same as is described above with respect to the second process.

The value of the predetermined exponent,  $k$ , is determined in step 72 (FIG. 6). The system noise level 70 is a matrix of elements corresponding to the elements of the captured Bayer image data, but representing system noise level. For example, a totally dark image could be captured in the sensor and loaded into matrix 70, or an imaging device could contain a group of light shielded pixels used to determine  $k$ . An exemplary way in which  $k$  may be determined is to average the noise value over all elements to determine an overall average, and then compute a variance to this average over all elements of the matrix. The variance becomes input to a look up table where the value of  $k$  is read as an output. In this way, different loads of the look up table can be used to emphasize edge clarity in low-noise situations and avoid enhancing noise while maintaining minimal color aliasing in high-noise situations.

After a full RGB image is recovered in block 118 (FIG. 6), the image in RGB space is converted to chroma space ( $C'$  and  $C''$  as discussed above) in block 120. In block 122, the chroma components (e.g.,  $C'$  and  $C''$ ) are filtered to remove false colors and high frequency spikes in a low pass chroma filter (e.g., such as a moving window of a 3 by 3 kernel of elements where the chroma value of the center point takes up the average value over the 3 by 3 kernel). Then, in block 124, the chroma space image is converted back into RGB space.

The processes described above for determining Red and Blue values require that Green values be determined for all pixels in the 3 by 3 kernel used to determine the Red values or the kernel used to determine the Blue values. As can be observed in FIG. 6, a one row and a one column delay is introduced in block 66. This permits Green values to

be determined one row and one column ahead of the determination of either Red or Blue values. Thus, when a 3 by 3 kernel is extracted to determine the Red values or to determine the Blue values, the entire kernel already has the Green value determined for each element of the 3 by 3 kernel. In this way, the process lends itself for implementation  
5 in a processor (be it general purpose or DSP) or an Application Specific Integrated Circuit (ASIC), but it could always be implemented in discrete circuits connected together. Regardless of the specific technology involved, the above described method is implemented in a computing machine of some form that includes a processor and a memory. The memory stores a first module for controlling the processor to perform a first process for a  
10 first pixel. The first process includes extracting a first kernel from a multi-color matrix, generating first variance weights from the first kernel, and generating a first color based on the first variance weights and adjacent pixel values of the first color. The memory further stores a second module for controlling the processor to perform a second process for the first pixel. The second process includes extracting a second kernel from the multi-color matrix,  
15 generating second variance offsets from the second kernel, and generating a second color based on the second variance offsets and an adjacent pixel of the second color. The processor may be a programmable processor that is controlled by programming modules stored on a separate computer readable media or it may be a fixed designed that is controlled by the timing generator of the processor.

20 In the event that the processes are implemented on a programmable computing machine (such as a general purpose processor or DSP), the details of above described method are included in program modules stored on a computer readable media. The computer readable media has stored thereon a plurality of modules for controlling a processor. The plurality of modules include first and second modules. The first module controls the  
25 processor to perform a first process for a first pixel. The first process includes extracting a first kernel from a multi-color matrix, generating first variance weights from the first kernel, and generating a first color based on the first variance weights and adjacent pixel values of the first color. The second module controls the processor to perform a second process for the first pixel. The second process includes extracting a second kernel from the multi-color matrix,  
30 generating second variance offsets from the second kernel, and generating a second color based on the second variance offsets and an adjacent pixel of the second color.

Having described preferred embodiments of a novel method and apparatus for reconstructing pixel color values (which are intended to be illustrative and not limiting), it is noted that modifications and variations can be made by persons skilled in the art in light of the above teachings. It is therefore to be understood that changes may be made in the particular embodiments of the invention disclosed which are within the scope and spirit of the invention as defined by the appended claims.

Having thus described the invention with the details and particularity required by the patent laws, what is claimed and desired protected by Letters Patent is set forth in the appended claims.